Brief Summary of Selected Substantive Portions of this Response

[0002] In a telephone discussion (on April 24, 2007), Examiners Traore and Myhre indicated that the references used to reject claim 15 did not disclose all of the features claimed therein. The Examiners also indicated that although the office action failed to state § 101 rejections, claim 1 and others would be rejected as directed to non-statutory subject matter.

[0003] This brief summary is not intended to represent the Applicant's full response to the Action. Rather, it is merely a brief summary of selected substantive portions of the response herein.

SPECIFICATION AMENDMENTS

Please amend the specification as follows:

Substitute the paragraph [0023] at page 6, with the following:

[0023] Fig. 1 illustrates an exemplary secure hash function method 100. The method 100 starts with a stage 102 which divides an input string into blocks of fixed length (as will be further discussed herein, for example, with respect to Figs. 2 and 4). A block function is then applied to the first input block (104). The block function will be further discussed below under the same title. If more input blocks remain for processing (106), the block function is applied to a next input block in accordance with select properties of the block function applied to the previous block (as discussed below, for example, with

reference to matrix graphs). In one implementation, the advanced encryption standard (AES) may be used as the inter-block function (as will be further discussed with reference to Fig. 4). If all input blocks have been processed, the hash value ~~if~~of the input is determined based on the result provided by the block function applied to the last input block (110).

Substitute the paragraph [0034] at page 9, with the following:

[0034] In one implementation, the block function is based on a walk on a ~~Caley~~Cayley graph (referred to herein as a "matrix graph") defined by the matrices. Let some $m=2^l$ (e.g., $m=2^{32}$), where m is the number of nodes in the matrix graph. Let $A=\{\sigma_1,...,\sigma_k\}$ be a set of generators (such as those shown in Fig. 3 and further discussed below), with $\sigma_i \in \square_m^{r\times r}$ for some r, that is, a $r \times r$ matrix over integers modulo m. In certain examples herein, the constructions shown use $r = 3$. Let the graph $G=G_A$ with vertex set $V = \square_m^{r\times r}$ and edges $\{(M,N) \mid N = \sigma_i, \sigma_i \in A\}$. Note G is a directed graph.

Substitute the paragraph [0046] at page 12, with the following:

[0046] In a stage 404, the initial matrix M is set to be the identity matrix (such as discussed with reference to the matrix I in the vertex set V). A stage 406

processes the data input blocks (e.g., as 9-bit blocks in the example discussed with reference to Fig. 3). Each block is used to index a matrix A in the table (408). As long as more input blocks remain for processing (410), a stage 412 updates M ($M \leftarrow A \cdot M$). Once all input blocks are processed (410), a stage 414 determines the final hash value (such as discussed with reference stage ~~310~~110 of Fig. ~~3~~1).

Substitute the paragraph [00109] at page 25, with the following:

[0046] Any number of program modules can be stored on the hard disk 616, magnetic disk 620, optical disk 624, ROM 612, and/or RAM 610, including by way of example, an operating system 626, one or more application programs 628, other program modules 630, and program data 632. Each of such operating system 626, one or more application programs 628, other program modules 630, and program data 632 (or some combination thereof) may implement all or part of the resident components that support the distributed file system. Example of program modules and data is discussed below with reference to Fig. ~~22~~6.